A18 ANALYTICS

Vizzify

Product Brief: BRD, PRD and TRD

Dashboard Design and Reporting Experience Platform

Vizzify helps analytics teams design clearer, more consistent, and more professional dashboards by turning the final reporting experience into a reusable, reviewable, and user-friendly product layer.

Prepared for product positioning, delivery planning, MVP definition, and implementation alignment

Version 1.0 | May 2026

1. Executive Overview

Vizzify is a dashboard design and reporting experience platform created by A18 Analytics to address a problem that is visible in almost every growing business intelligence environment. Organizations invest in data warehouses, semantic models, reporting tools, and analytical talent, yet the final experience through which users consume insight is often left to individual preference. One analyst chooses a colour system, another chooses a different layout, a third introduces a different naming style, and over time the reporting estate becomes harder to read, harder to govern, and harder to scale. The issue is not always that the data is wrong. Very often, the issue is that the experience through which the data is presented does not give users enough confidence, continuity, or clarity.

The product is therefore positioned around a simple but important idea: the dashboard is not decoration; it is the last mile of analytics delivery. Most business users never see the ingestion pipeline, the transformation logic, the star schema, or the semantic model. They see the page in front of them, and they make decisions based on whether that page helps them understand what matters. Vizzify exists to make that final layer more deliberate. It helps teams create reusable dashboard patterns, apply visual standards, review reporting experiences, and produce dashboards that feel consistent enough for enterprise use while remaining soft, friendly, and understandable for end users.

This document organizes Vizzify into three connected planning views. The Business Requirements Document explains the commercial problem, the intended users, the operating context, and the business outcomes the product should support. The Product Requirements Document translates those business needs into user journeys, functional requirements, non-functional expectations, and release priorities. The Technical Requirements Document then describes the architecture, data model, integrations, security principles, and implementation approach required to deliver the product in a way that can grow from a focused minimum viable product into a broader analytics experience platform.

Product positioning statement

Vizzify helps analytics teams design dashboards that are clearer, more consistent, and easier for users to trust by providing reusable layouts, visual style guidance, brand-friendly reporting patterns, and structured review workflows for the reporting experience.

Area	Primary intent	Business value	Delivery focus
BRD	Define the business case and operating problem.	Improves reporting trust, adoption, and delivery consistency.	Market, users, process, success metrics.
PRD	Translate business need into product behaviour.	Clarifies what the platform must help users accomplish.	Personas, journeys, features, release scope.
TRD	Define the technical delivery model.	Ensures the product is secure, scalable, and maintainable.	Architecture, data, APIs, integrations, deployment.

2. Business Requirements Document

2.1 Business context and opportunity

Business intelligence has matured considerably in most organizations, but the design discipline surrounding dashboards has not always kept pace. Many teams now have certified datasets, governed semantic models, automated refreshes, and cloud reporting environments, yet the visible reporting layer still behaves like a collection of personal design decisions. This weakens adoption because users do not experience analytics as a coherent product. They experience it as a set of pages that may or may not look related, may or may not be easy to read, and may or may not present measures in a way that supports decision-making.

This creates a clear opportunity for Vizzify. The product sits between analytics delivery, user experience, and report governance. It does not need to replace existing BI tools; instead, it strengthens the way teams design, standardize, and review reports within the environments they already use. In the immediate sense, Vizzify can help teams produce cleaner dashboards faster. In the broader strategic sense, it can help organizations treat reporting as a managed experience layer rather than a loose collection of report files.

2.2 Problem statement

The central business problem is that dashboard quality is often inconsistent because organizations lack a practical system for managing the design of their reporting experience. Reporting teams may have strong technical skill, but without shared visual rules, reusable patterns, and lightweight review workflows, dashboards can become crowded, visually disconnected, and difficult for business users to interpret. This problem is especially visible in environments where multiple teams, consultants, departments, or client projects produce reports under time pressure.

The consequence is not merely aesthetic. Inconsistent dashboard design increases the cognitive load placed on users, slows down interpretation, weakens confidence in the reporting environment, and creates unnecessary rework for analytics teams. When every report requires fresh decisions about layout, colour, chart placement, filter design, and KPI hierarchy, delivery becomes slower and less predictable. When every dashboard looks different, users must relearn the interface each time they

open a report. Vizzify is intended to reduce this friction by introducing a softer, repeatable structure to the reporting experience.

2.3 Business objectives

The first business objective is to improve the clarity and consistency of dashboards produced by analytics teams. Vizzify should make it easier for report builders to begin from approved design patterns rather than from a blank page, and it should help organizations establish a recognizable visual language across dashboards. This does not mean every report must look identical. Rather, it means that users should sense continuity in layout, emphasis, navigation, and visual behaviour.

The second objective is to reduce report delivery effort by making reusable dashboard structures available to teams. If a reporting team repeatedly builds executive summaries, operational monitoring pages, customer dashboards, finance packs, or performance reviews, Vizzify should help them reuse proven arrangements instead of redesigning each one from the beginning. This improves delivery discipline while still allowing room for adaptation.

The third objective is to support trust in analytics by improving the experience through which insight is consumed. A well-designed dashboard should help users know where to look, what matters most, and how the presented measures relate to their decisions. Vizzify should therefore be assessed not only by how attractive its outputs are, but by whether it improves readability, consistency, user confidence, and reporting adoption.

2.4 Target users and stakeholders

Vizzify is designed for several overlapping groups. The primary users are BI developers, data analysts, reporting teams, and analytics consultants who need to create dashboards quickly without sacrificing quality. These users need practical templates, visual guidance, theme consistency, and review support that can improve their daily workflow without slowing them down. They are not looking for another abstract governance document; they need tools that meet them inside the practical reality of report delivery.

The secondary users are analytics managers, data product owners, and business intelligence leads who are responsible for report quality across teams. These stakeholders need confidence that dashboards produced by different people still meet a shared standard. They also need visibility into where design inconsistencies are appearing and where the reporting experience may require improvement. The end beneficiaries are business users, executives, clients, and operational teams who depend on dashboards to make decisions. For them, the value of Vizzify is felt through clearer pages, more predictable layouts, and a softer experience that reduces confusion.

2.5 Scope of the business capability

At launch, Vizzify should focus on the reporting experience layer rather than attempting to become a full BI platform. Its business capability should include dashboard pattern management, visual standard guidance, theme support, report experience review, brand alignment, and documentation of approved design rules. The product should be able to support teams that are using Power BI or similar dashboarding environments, but its language and positioning should remain broader than a single tool so that the product is not trapped inside one vendor identity.

The business scope should also include a service-led pathway for A18 Analytics. Vizzify can operate as both a product and an implementation offering. A18 can use it to support dashboard audits, BI design system development, report redesign engagements, consulting delivery packs, client-facing analytics templates, and internal analytics modernization projects. This makes Vizzify commercially useful even before every future software feature is fully automated.

2.6 Success metrics

The success of Vizzify should be measured by the extent to which it improves reporting delivery and user experience. A beautiful dashboard is useful, but the deeper measure is whether the dashboard becomes easier to build, easier to review, easier to reuse, and easier to understand. Therefore, the product should track both adoption metrics and quality indicators.

Metric group	Example measures	Why it matters
Adoption	Number of teams using templates, active users, projects created, repeat usage.	Shows whether the product is becoming part of reporting work.
Delivery efficiency	Time saved in dashboard setup, template reuse rate, reduction in redesign cycles.	Shows whether Vizzify reduces repetitive design effort.
Design consistency	Reports passing layout checks, theme compliance score, pattern adherence.	Shows whether the reporting estate is becoming more coherent.
User experience	Readability ratings, stakeholder review outcomes, business user feedback.	Shows whether dashboards are easier to understand and trust.
Commercial value	Consulting projects supported, clients onboarded, conversion from audit to implementation.	Shows whether Vizzify strengthens A18 revenue opportunities.

3. Product Requirements Document

3.1 Product vision

The product vision for Vizzify is to become the friendly design layer that helps analytics teams turn dashboard creation into a more consistent, reusable, and user-centered process. The experience should feel approachable to report builders who want practical help, while still being structured enough for managers who care about quality, brand alignment, and enterprise reporting discipline. The product should avoid the stiffness of a governance portal and instead present itself as a design companion for teams that want better dashboards without unnecessary complexity.

In practical terms, Vizzify should allow a user to choose or create a dashboard pattern, apply a visual style, review design guidance, and prepare a reporting experience that feels polished before it is delivered. The product should not overwhelm users with heavy terminology. Its interface should speak in the language of clarity, layout, consistency, readability, and reuse. Governance can exist underneath, but the surface should feel soft enough for end users and report builders to accept it as a natural part of their workflow.

3.2 Personas

The BI developer is the first core persona. This user is responsible for building dashboards and often works under delivery pressure. They need Vizzify to give them reusable structures, design guidance, and practical review support without creating a long approval process. They will value anything that helps them move faster while making their work look more professional.

The analytics manager is the second core persona. This user is responsible for the quality and consistency of reporting across a team or function. They need Vizzify to make standards visible, reusable, and easier to apply. Their concern is not only whether one dashboard looks good, but whether the entire reporting environment feels controlled, credible, and aligned with the organization.

The consultant or client delivery lead is the third persona. This user needs to produce dashboards for different clients or projects, often while adapting to different brands and business contexts. They need a way to create polished reporting experiences repeatedly without starting from zero each time. Vizzify should help them create credibility through structure, speed, and visual professionalism.

The business user is the final and most important beneficiary. This user may never configure Vizzify directly, but they experience its value through dashboards that are less crowded, better organized, and easier to interpret. For this user, success means spending less time trying to understand the interface and more time using the insight.

3.3 User journeys

A typical user journey begins when a report builder starts a new dashboard project and chooses a design objective. The objective may be an executive overview, an operational monitoring page, a financial performance report, a client-facing dashboard, or a KPI scorecard. Vizzify should then guide the user toward an appropriate layout pattern, explain the purpose of that pattern, and allow the user to adapt it to the reporting context. This gives the builder a starting point that is more intelligent than a blank canvas.

A second journey involves applying a visual style or brand system. The user should be able to select approved colours, chart behaviours, typography preferences, and layout rules that align with either an organizational brand or a project-specific design language. This should not feel like a technical configuration exercise. It should feel like choosing a coherent design direction that makes the dashboard more readable.

A third journey involves reviewing the dashboard experience before delivery. Vizzify should help the user assess whether the page is overloaded, whether important metrics are visible, whether colours are used consistently, whether filters are placed sensibly, and whether the overall visual hierarchy supports comprehension. The review should be advisory and constructive, not punitive. It should help users improve the work rather than make them feel blocked by a governance process.

3.4 Functional requirements

The functional requirements for Vizzify should be organized around creation, standardization, review, and reuse. The creation layer should allow users to create projects, define dashboard types, select layout patterns, and associate a reporting context with each project. The standardization layer should allow users to manage themes, colour palettes, visual rules, layout guidelines, and reusable component patterns such as KPI cards, chart groups, filter sections, and navigation areas.

The review layer should provide a structured way to assess dashboard quality. At minimum, it should support checks for page density, inconsistent colour use, missing titles, weak visual hierarchy, chart overload, unclear KPI placement, filter complexity, and brand misalignment. These checks may begin as rule-based guidance and later evolve into AI-assisted recommendations that interpret screenshots, metadata, or report configuration files.

The reuse layer should allow teams to save approved patterns, duplicate them for new projects, organize them by use case, and document why a pattern exists. This is especially important because Vizzify should not only help teams create one better dashboard; it should help them build an internal library of better dashboard decisions.

3.5 Feature backlog and release scope

Release	Feature area	Description	Priority
MVP	Project workspace	Create dashboard design projects and assign a report type, audience, and delivery context.	**High**
MVP	Layout library	Provide reusable dashboard structures for common reporting scenarios.	**High**
MVP	Theme guidance	Manage brand colours, chart defaults, typography notes, and spacing principles.	**High**
MVP	Design checklist	Offer a simple review workflow for readability, hierarchy, clutter, and consistency.	**High**
V1	Pattern library	Save approved dashboard patterns and reuse them across teams or clients.	**High**
V1	Export package	Export design briefs, style notes, and implementation guidance for developers.	**Medium**
V2	Report inspection	Analyze report files, metadata, or screenshots for potential design issues.	**Medium**
V2	AI suggestions	Generate improvement recommendations based on dashboard purpose and audience.	**Medium**

3.6 Non-functional requirements

Vizzify must feel fast, calm, and responsive because the product is meant to reduce friction in reporting work. Users should not feel as if they have opened an administrative compliance system. The interface should be clean, bright, and soft, with enough whitespace to reinforce the product’s purpose. Performance should support quick movement between projects, patterns, guidance, and review screens. Accessibility should also be treated as part of product quality because dashboard design itself is inseparable from readability and inclusion.

The platform should also support secure user management, role-based access, and organization-level separation of projects and design assets. A consulting user should be able to separate client work, while an enterprise user should be able to keep internal design systems controlled. Reliability matters because design standards become more valuable as they are reused; users must trust that saved patterns, themes, and guidance will remain available and versioned over time.

4. Technical Requirements Document

4.1 Technical architecture overview

The recommended technical architecture for Vizzify is a modern web application with a clear separation between the front-end experience, the application service layer, the data persistence layer, and any optional AI or report-inspection services. The front end should be built as a responsive, component-driven application that presents projects, layout libraries, design guidance, review results, and exportable documentation in a clean workflow. A React or Next.js application is suitable because Vizzify will benefit from reusable interface components, routing, and a polished product experience.

The back end should expose a structured API that manages users, organizations, projects, layout patterns, theme assets, checklist rules, review outputs, and audit history. FastAPI or a comparable framework would be appropriate because it supports clean API design, typed schemas, and future integration with AI-assisted services. The database should store product configuration, design-system assets, review records, and versioned templates. PostgreSQL is a strong default because it supports relational structure, JSON fields for flexible design configuration, and future analytics on usage patterns.

4.2 Logical components

Component	Purpose	Key responsibilities
Web application	Provides the user-facing product experience.	Project creation, layout selection, theme setup, review workflow, exports.
API service	Coordinates business logic and secure access.	Authentication, organization access, CRUD operations, checklist execution, audit events.
Design repository	Stores reusable visual standards and patterns.	Layouts, components, palettes, typography rules, page structures, version history.
Review engine	Evaluates dashboard design against rules.	Readability checks, consistency checks, density checks, recommendation generation.
Export service	Produces handoff assets for teams.	PDF briefs, HTML snippets, design notes, implementation packages.
Inspection service	Supports future analysis of reports or screenshots.	Metadata parsing, screenshot analysis, AI-assisted recommendations.

4.3 Data model

The data model should be simple enough for an MVP but structured enough to support growth. At the highest level, Vizzify should manage organizations, users, projects, design systems, layout patterns, theme assets, checklist rules, review sessions, review findings, and export artifacts. This structure allows the product to support both individual project work and organization-level reuse.

An organization should own users, projects, and design systems. A design system should contain one or more themes, layout patterns, visual components, and review rules. A project should reference the relevant design system, define its audience and dashboard type, and record review outcomes as the dashboard evolves. Review findings should be stored in a way that allows teams to see common issues over time, because this can later become a valuable management view for reporting quality.

4.4 API requirements

The API should support clear resources rather than a large collection of loosely named endpoints. User and organization endpoints should manage access. Project endpoints should create, update, list, and archive dashboard design projects. Pattern endpoints should expose approved dashboard structures, while theme endpoints should manage colours, typography notes, and visual rules. Review endpoints should run checklist evaluations and store findings. Export endpoints should generate design handoff documents that can be shared with report builders or stakeholders.

From an implementation perspective, the API should return typed JSON responses, validate incoming requests carefully, and maintain audit events for meaningful actions such as creating a design system, approving a pattern, running a review, or exporting a project brief. This is important because Vizzify is partly a governance product even when its user interface feels soft and friendly. The governance should not dominate the user experience, but it should exist underneath in a traceable way.

4.5 Integration considerations

The first version of Vizzify can function without deep integration into Power BI or another BI platform, because much of the value can be delivered through layout libraries, design guidance, checklists, and exportable design briefs. However, the architecture should leave room for future integrations. Over time, the platform may inspect Power BI theme JSON files, analyze report screenshots, parse report metadata, or generate starter assets that developers can use inside their BI environment.

This staged approach is important because it avoids turning the MVP into an integration-heavy engineering project before the core product value is proven. The initial product should help users design and review better dashboard experiences. Later versions can automate more of the inspection, validation, and handoff process as the product matures.

4.6 Security, roles, and governance

Vizzify should support role-based access because design standards can be both collaborative and controlled. An administrator should manage organization settings, design systems, and approved patterns. A builder should create projects, use templates, and run reviews. A reviewer should comment on design quality and approve patterns where appropriate. A viewer should access published design guidance without changing it. These roles allow the product to remain flexible while protecting the integrity of reusable standards.

The product should also maintain an audit trail for important changes. When a theme is updated, when a pattern is approved, when a review is completed, or when a design brief is exported, the system should know who performed the action and when it occurred. This supports accountability without making the product feel bureaucratic. The principle is simple: make the user experience soft, but make the underlying product trustworthy.

4.7 Deployment and operating model

For early development, Vizzify can be deployed as a containerized web application with separate services for the front end, API, database, and optional background worker. A simple Docker Compose setup is adequate for local development and demonstration. For production, the front end can be hosted on a managed platform, while the API and database can be deployed using cloud infrastructure appropriate to the client or A18 operating environment. The product should be designed so that it can later support multi-tenant SaaS deployment or dedicated client environments.

Observability should include application logs, API request tracking, error monitoring, and basic usage analytics. Since Vizzify is itself concerned with quality, its operating model should reflect the same discipline. The team should know which features are being used, which review checks are generating the most findings, and where users are dropping off in the design workflow. This feedback will guide product improvement and help A18 understand how teams actually use the platform.

5. Implementation Roadmap

The implementation roadmap should begin with a focused MVP that proves the core value proposition: helping teams create and review dashboard designs more consistently. The first build should include a polished landing experience, a project workspace, a small but strong library of layout patterns, theme guidance, a dashboard design checklist, and an exportable product brief or design handoff. This is enough to demonstrate the practical usefulness of Vizzify without waiting for advanced integrations.

The next release should strengthen reuse and collaboration. Users should be able to save approved patterns, organize them by use case, duplicate them across projects, and document design decisions. This release should also introduce better role separation, review history, and management views that show how teams are applying the standards. Once these foundations are stable, later releases can introduce richer inspection capabilities, including screenshot review, Power BI theme parsing, metadata extraction, and AI-assisted recommendations.

Phase	Primary outcome	Core deliverables	Decision gate
Phase 1	Validate the product experience.	Landing page, project workspace, initial layout library, checklist workflow.	Users can create a useful design brief.
Phase 2	Make standards reusable.	Pattern library, design-system management, theme assets, review history.	Teams can reuse approved dashboard decisions.
Phase 3	Introduce automation.	Report inspection, screenshot review, AI recommendations, richer exports.	The platform can detect and explain design issues.
Phase 4	Scale commercially.	Multi-tenant controls, client workspaces, analytics dashboard, service packaging.	A18 can use Vizzify as both product and consulting accelerator.

6. Risks, Assumptions, and Controls

The main product risk is that Vizzify could be perceived as a cosmetic tool rather than a practical reporting experience platform. This risk should be managed through language, workflow, and examples. The product should avoid speaking only about beauty, colours, and polish. It should consistently connect design quality to readability, consistency, adoption, and trust. In other words, the product should make the case that dashboard design is a decision-support capability, not a decorative layer.

A second risk is overbuilding too early. Deep BI integrations, AI-assisted analysis, and automated report inspection are attractive, but they should not delay the first release if the core product can deliver value through structured guidance, reusable patterns, and review workflows. The recommended control is to keep the MVP grounded in the work users already do while designing the architecture so that richer automation can be introduced later.

A third risk is user resistance. Report builders may not want to feel controlled by a design system, especially if they are used to working independently. Vizzify should therefore be framed as an enablement tool rather than a policing tool. Its recommendations should be constructive, its interface should be calm, and its checklists should help builders improve quality without making them feel that creativity has been removed.

7. Conclusion

Vizzify occupies an important space in the analytics value chain because it focuses on the point where data finally becomes visible to the people who must use it. Organizations often speak about data governance, platform modernization, semantic modelling, and AI readiness, yet the dashboard experience remains the place where many users decide whether they trust the work. If the dashboard is crowded, inconsistent, or visually confusing, the technical strength behind it may not be fully appreciated. Vizzify responds to that reality by treating the reporting experience as a product layer that deserves structure, reuse, and review.

The business case is clear. Vizzify can help teams reduce inconsistent dashboard styling, speed up report creation, improve user readability, and create a more professional analytics presence. The product case is also clear. Users need reusable layouts, visual standards, design guidance, review workflows, and exportable handoff assets that make their work easier without making the process feel heavy. The technical case is equally practical. A modern web application with a strong API layer, PostgreSQL-backed design repository, role-based access, and staged integration strategy can support the product from MVP to a more advanced SaaS or consulting-enabled platform.

For A18 Analytics, Vizzify is more than a standalone product. It is a statement about how analytics should be delivered. Data products should not only be technically functional; they should be understandable, reusable, and trustworthy at the point of use. Vizzify gives A18 a soft, front-facing product that speaks to end users while still carrying serious enterprise logic beneath the surface. That balance is the product's strength: it makes dashboard design feel approachable, but it treats the reporting experience with the discipline it deserves.

> **Recommended next step:** use this brief as the foundation for the Vizzify product page, MVP backlog, interface copy, and Cursor implementation prompt, while keeping the public-facing language softer than the internal governance logic.